Title

stata.com

power, table — Produce table of results from the power command

Description

`power, table` displays results in a tabular format. `table` is implied if any of the `power` command's arguments or options contain more than one element. The `table` option is useful if you are producing graphs and would like to see the table as well or if you are producing results one case at a time using a loop and wish to display results in a table. The `notable` option suppresses table results; it is implied with the graphical output of `power, graph`; see [PSS] **power, graph**.

Menu

Statistics > Power and sample size

Syntax

Produce default table

```
power ..., table ...
```

Suppress table

```
power ..., notable ...
```

Produce custom table

```
power ..., table([ colspec ] [ , tableopts ]) ...
```

where *colspec* is

column[:*label*] [*column*[:*label*] [...]]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

tableopts	Description
Table	
`add`	add *columns* to the default table
`labels(`*labspec*`)`	change default labels for specified columns; default labels are column names
`widths(`*widthspec*`)`	change default column widths; default is specific to each column
`formats(`*fmtspec*`)`	change default column formats; default is specific to each column
`noformat`	do not use default column formats
`separator(#)`	draw a horizontal separator line every # lines; default is `separator(0)`, meaning no separator lines
`divider`	draw divider lines between columns
`byrow`	display rows as computations are performed; seldom used
`noheader`	suppress table header; seldom used
`continue`	draw a continuation border in the table output; seldom used

`noheader` and `continue` are not shown in the dialog box.

column	Description
`alpha`	significance level
`power`	power
`beta`	type II error probability
`N`	total number of subjects
`N1`	number of subjects in the control group
`N2`	number of subjects in the experimental group
`nratio`	ratio of sample sizes, experimental to control
`K`	number of clusters
`K1`	number of clusters in the control group
`K2`	number of clusters in the experimental group
`kratio`	ratio of numbers of clusters, experimental to control
`M`	cluster size
`M1`	cluster size in the control group
`M2`	cluster size in the experimental group
`mratio`	ratio of cluster sizes, experimental to control
`delta`	effect size
`target`	target parameter
`_all`	display all supported columns
method`_columns`	columns specific to the method specified with `power`

By default, the following columns are displayed:

`alpha` and `power` are always displayed;
`N` is always displayed except for two-sample methods in a CRD;
`N1` and `N2` are displayed for two-sample methods except for a CRD;
`kratio` and `mratio` are available for two-sample methods in a CRD;
`delta` is displayed when defined by the method;
additional columns specific to each `power` method may be displayed.

Suboptions

The following are suboptions within the `table()` option of the `power` command.

Table

`add` requests that the columns specified in *colspec* be added to the default table. The columns are added to the end of the table.

`labels(`*labspec*`)` specifies the labels to be used in the table for the specified columns. *labspec* is

column "*label*" [*column* "*label*" [...]]

`labels()` takes precedence over the specification of column labels in *colspec*.

`widths(`*widthspec*`)` specifies column widths. The default values are the widths of the default column formats plus one. If the `noformat` option is used, the default for each column is nine. The column widths are adjusted to accommodate longer column labels and larger format widths. *widthspec* is either a list of values including missing values (*numlist*) or

column # [*column* # [...]]

For the value-list specification, the number of specified values may not exceed the number of columns in the table. A missing value (`.`) may be specified for any column to indicate the default width. If fewer widths are specified than the number of columns in the table, the last width specified is used for the remaining columns.

The alternative column-list specification provides a way to change widths of specific columns.

`formats(`*fmtspec*`)` specifies column formats. The default is `%7.0gc` for integer-valued columns and `%7.4g` for real-valued columns. *fmtspec* is either a string value-list of formats that may include empty strings or a column list:

column "*fmt*" [*column* "*fmt*" [...]]

For the value-list specification, the number of specified values may not exceed the number of columns in the table. An empty string (`""`) may be specified for any column to indicate the default format. If fewer formats are specified than the number of columns in the table, the last format specified is used for the remaining columns.

The alternative column-list specification provides a way to change formats of specific columns.

`noformat` requests that the default formats not be applied to the column values. If this suboption is specified, the column values are based on the column width.

`separator(#)` specifies how often separator lines should be drawn between rows of the table. The default is `separator(0)`, meaning that no separator lines should be displayed.

`divider` specifies that divider lines be drawn between columns. The default is no dividers.

`byrow` specifies that table rows be displayed as computations are performed. By default, the table is displayed after all computations are performed. This suboption may be useful when the computation of each row of the table takes a long time.

The following suboptions are available but are not shown in the dialog box:

`noheader` prevents the table header from displaying. This suboption is useful when the command is issued repeatedly, such as within a loop.

`continue` draws a continuation border at the bottom of the table. This suboption is useful when the command is issued repeatedly, such as within a loop.

Remarks and examples

stata.com

Remarks are presented under the following headings:

`power, table` displays results from the `power` command in a table. This is useful for sensitivity analysis, which investigates the effect of varying study parameters on power, sample size, or other components of the study. The true values of study parameters are usually unknown. PSS analysis uses best guesses for these values. It is important to evaluate the sensitivity of the computed power or sample size to the chosen values of study parameters. For example, to evaluate variability of power values, you can compute powers for various ranges of values for the parameters of interest and display the resulting powers in a table or plot them on a graph (see [PSS] **power, graph**).

Using power, table

If you specify the `table` option or include more than one element in command arguments or in options allowing multiple values, the `power` command displays results in a tabular form. If desired, you can suppress the table by specifying the `notable` option. The `table` option is useful if you are producing graphical output or if you are producing results one case at a time, such as within a loop, and wish to display results in a table; see example 4 below.

Each method specified with the `power` command has its own default table. Among the columns that are always included in the default table are significance level (`alpha`), power (`power`), total sample size (`N`)—except for two-sample methods in a cluster randomized design (CRD)—and effect size (`delta`)—if effect size is defined by the method.

Depending on the method and study design, additional columns are also included by default. One-sample methods in a CRD include the number of clusters (`K`) and cluster size (`M`). Two-sample methods in an individual-level design include the sample sizes of the control and experimental groups (`N1` and `N2`). Two-sample methods in a CRD include the numbers of clusters of the control and experimental groups (`K1` and `K2`) and the cluster sizes of the two groups (`M1` and `M2`).

You can build your own table by specifying the columns and, optionally, their labels in the `table()` option. You can also add columns to the default table by specifying `add` within `power`'s `table()` option. The columns are displayed in the order they are specified. Each method provides its own list of supported columns; see the description of the `table()` option for each method. You can further customize the table by specifying various suboptions within `power`'s `table()` option.

The default column labels are the column names. You can provide your own column labels in *colspec* or by specifying `table()`'s suboption `labels()`. Labels containing spaces should be enclosed in quotes, and labels containing quotes should be enclosed in compound quotes. The `labels()` suboption is useful for changing the labels of existing columns; see example 2 below for details.

The default formats are `%7.4g` for real-valued columns and `%7.0gc` for integer-valued columns. If the `noformat` suboption is specified, the default column widths are nine characters. You can use `formats()` to change the default column formats and `widths()` to change the default column widths. The `formats()` and `widths()` suboptions provide two alternative specifications, a value-list specification or a column-list specification. The value-list specification accepts a list of values—strings for formats and numbers for widths—corresponding to each column of the displayed table. Empty strings (`""`) for formats and missing values (`.`) for widths are allowed and denote the default values. It is an error to specify more values than the number of displayed columns. If fewer values are specified,

then the last value specified is used for the remaining columns. The column-list specification includes a list of pairs containing a column name followed by the corresponding value of the format or width. This specification is useful if you want to modify the formats or the widths of only selected columns. For column labels or formats exceeding the default column width, the widths of the respective columns are adjusted to accommodate the column labels and the specified formats.

If you specify the `noformat` suboption, the default formats are ignored, and the format of a column is determined by the column width: if the column width is #, the displayed format is `%(# - 2).#g`. For example, if the column width is 9, the displayed format is `%7.0g`.

You may further customize the look of the table by using `separator(#)` to include separator lines after every # lines and by using the `divider` suboption to include divider lines between columns.

The `noheader` and `continue` suboptions are useful when you are building your own table within a loop; see example 4 in *Custom tables*.

In what follows, we demonstrate the default and custom tables of the results from power and sample-size analysis for a two-sided 5%-level one-sample t test comparing the population mean with a hypothesized value; see [PSS] **power onemean**.

Default tables

If there is only one set of results, the `power` command displays those results as text. When the `power` command has multiple sets of results, they are automatically displayed in a table. You can also specify the `table` option at any time to request that results be displayed in a table.

The displayed columns are specific to the chosen method of analysis and to the options specified with the command. The columns that always appear in the table include the significance level (`alpha`), power (`power`), and total sample size (`N`). If the concept of effect size is defined for the chosen method, the effect size (`delta`) is also displayed in the default table.

▷ Example 1: Default tables from power onemean

Suppose we want to explore the requirements on the sample size for a one-sample mean comparison test to detect means of different magnitudes. For simplicity, we consider only two target mean values, 1 and 2, and keep all other study parameters of `power onemean` at their default values: power at 80%, two-sided significance level at 0.05, and the estimate of the population standard deviation at 1. We use a zero null value of the mean. See [PSS] **power onemean** for details.

```
. power onemean 0 (1 2)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd
.05	.8	10	1	0	1	1
.05	.8	5	2	0	2	1

As we mentioned earlier, the `alpha`, `power`, `N`, and `delta` columns are usually displayed in the default table. The `power onemean` command additionally displays columns containing the null mean, the alternative mean, and the standard deviation.

If we need to account for finite population, we can specify the `fpc()` option with `power onemean`. The default table will contain an additional column, `fpc`.

```
. power onemean 0 (1 2), fpc(500)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd	fpc
.05	.8	10	1	0	1	1	500
.05	.8	5	2	0	2	1	500

If there is only one set of results, the `power` command displays those results as text. If desired, you can request a table by specifying the `table` option.

```
. power onemean 0 1, table
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd
.05	.8	10	1	0	1	1

◁

Modifying default tables

We can modify labels, widths, and formats of the default columns by specifying the corresponding suboptions within the `table()` option. We can also add columns to the default table by using `table()`'s suboption `add`.

▷ Example 2: Modifying default tables from power onemean

We can change the default labels of all or selected columns by using the `labels()` suboption within `power`'s `table()` option. For example, we can change the labels of the sample-size columns and standard deviation columns of the first table in example 1 to "Sample size" and "Std. Dev.", respectively.

```
. power onemean 0 (1 2), table(, labels(N "Sample size" sd "Std. Dev."))
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

alpha	power	Sample size	delta	m0	ma	Std. Dev.
.05	.8	10	1	0	1	1
.05	.8	5	2	0	2	1

We can also change default column formats and widths by using the `formats()` and `widths()` suboptions.

```
. power onemean 0 (1 2), n(5) table(, labels(N "Sample size" sd "Std. Dev.")
> widths(N 14 sd 14) formats(power "%7.5f"))

Estimated power for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

alpha	power	Sample size	delta	m0	ma	Std. Dev.
.05	0.40139	5	1	0	1	1
.05	0.90888	5	2	0	2	1

For this table, we switched from a sample-size determination to power determination by specifying the `n()` option. We changed the default column widths of the sample-size columns and standard deviation columns to 14. We also changed the default `%7.4g` format of the power column to `%7.5f`.

We can add columns to the default table by listing the names of the columns in the `table()` option and specifying its suboption `add`.

```
. power onemean 0 (1 2), table(diff, add)

Performing iteration ...

Estimated sample size for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd	diff
.05	.8	10	1	0	1	1	1
.05	.8	5	2	0	2	1	2

We added the column `diff`, which contains the difference between the alternative and hypothesized values of the mean, to the default table produced by `power onemean`.

◁

Custom tables

We can use the `table()` option to build custom tables, which contain the columns you want in the order you want. You can also build a table within a `foreach` or `forvalues` loop as we demonstrate in example 4 below. This is useful in the case when you want to obtain multiple sets of results over parameters of the `power` command that do not allow the *numlist* specification.

Example 3: Producing custom tables

As an example of a custom table, we produce a table containing only four columns: significance level, power, sample size, and effect size.

```
. power onemean 0 (1 2), table(alpha:"Significance level"
> power:Power N:"Sample size" delta:"Effect size", widths(. 15))
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

Significance level	Power	Sample size	Effect size
.05	.8	10	1
.05	.8	5	2

To improve the look of the table, we also specified the `widths(. 15)` suboption to increase the column widths of the last 3 columns to 15, leaving the width of the first column, the significance level, at its default value.

We can use the `_all` qualifier to request that all table columns supported by the method be displayed.

```
. power onemean 0 (1 2), table(_all)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

alpha	power	beta	N	delta	m0	ma	diff	sd
.05	.8	.2	10	1	0	1	1	1
.05	.8	.2	5	2	0	2	2	1

Example 4: Building table using a loop

Some options of `power` commands may not allow the *numlist* specification. In this case, you can build a table manually by using a loop such as `foreach` (see [P] **foreach**) or `forvalues` (see [P] **forvalues**) loop. One way to do this is to write a program that loops over parameters of interest. We demonstrate a program that loops over varying values of the alternative mean of `power onemean`. You can easily adapt this program to meet your needs.

```
program dotable
        args ma
        numlist "`ma'"                  // expand the numeric list in macro ma
        local ma "`r(numlist)'"
        local nvals : list sizeof ma
        local i 1
        foreach val of local ma {       // loop over numeric values in ma
                if (`i'==1) {
                        power onemean 0 `val', table(, continue)
                }
                else if (`i'<`nvals') {
                        power onemean 0 `val', table(, noheader continue) notitle
                }
                else {
                        power onemean 0 `val', table(, noheader) notitle
                }
                local ++i
        }
end
```

The `dotable` program accepts one argument, `ma`, which may contain one or more numeric values of the alternative mean specified as a *numlist*. The program uses combinations of `continue`, `noheader`, and `notitle` to display a table. The first call to `power onemean` requests that the table be displayed without the bottom line by specifying the `continue` suboption within `table()`. The subsequent calls (except the last) specify the `continue` suboption, the `notitle` option with `power onemean`, and `noheader` within the `table()` option to request that neither the output before the table nor the table header be displayed. The last call omits the `continue` suboption so that the bottom line is displayed.

As a result, we obtain the following table:

```
. dotable "1(1)4"
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
Ho: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd
.05	.8	10	1	0	1	1
.05	.8	5	2	0	2	1
.05	.8	4	3	0	3	1
.05	.8	3	4	0	4	1

◁

Stored results

`power, table` stores the following in `r()` in addition to other results stored by `power`:

Scalars

`r(separator)`	number of lines between separator lines in the table
`r(divider)`	`1` if `divider` is requested in the table; `0` otherwise

Macros

`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Also see

[PSS] **power** — Power and sample-size analysis for hypothesis tests

[PSS] **power, graph** — Graph results from the power command